Project 3

CS 475

Andrew Johnson

Write Up

I ran this this project on my personal machine. First time using it for a project in this class. I used g++. I remember hearing about using the Xeon Phi but I’m not sure about it and I like the use of Pragma, so I decided not to use Xeon Phi, still need to study up on it. Used up to 8 threads and up to 16 NUM.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | FIX #1 | | | | FIX #2 | | | |
|  | Threads | | | | Threads | | | |
| NUM | 1 | 2 | 4 | 8 | 1 | 2 | 4 | 8 |
| 0 | 32.3884 | 21.2069 | 18.643 | 18.4491 | 34.3947 | 60.2753 | 103.685 | 95.0553 |
| 1 | 35.07 | 21.5146 | 18.9978 | 18.9462 | 31.4988 | 60.6648 | 91.8025 | 88.3615 |
| 2 | 32.3727 | 30.7336 | 25.0836 | 22.5273 | 35.0762 | 59.7313 | 101.258 | 102.719 |
| 3 | 34.0742 | 59.5413 | 40.2898 | 38.1387 | 35.0786 | 66.6845 | 103.1 | 102.557 |
| 4 | 34.0726 | 66.7354 | 37.7572 | 37.782 | 35.0789 | 63.5273 | 102.234 | 101.912 |
| 5 | 35.0725 | 60.7943 | 37.9033 | 38.392 | 33.0348 | 59.4386 | 101.976 | 102.063 |
| 6 | 34.4015 | 60.1138 | 37.9384 | 38.0468 | 34.0649 | 64.7578 | 103.385 | 84.0855 |
| 7 | 34.401 | 65.1904 | 29.7119 | 28.8034 | 33.61 | 65.1093 | 102.916 | 107.811 |
| 8 | 32.4631 | 66.7568 | 31.0881 | 28.4657 | 32.1739 | 63.9279 | 103.089 | 104.796 |
| 9 | 34.7607 | 60.7737 | 29.2485 | 29.1355 | 35.0788 | 63.5375 | 103.197 | 107.466 |
| 10 | 33.9478 | 69.1342 | 28.0729 | 27.8339 | 34.4016 | 66.7765 | 107.438 | 90.1571 |
| 11 | 34.0513 | 63.7171 | 30.1886 | 30.006 | 35.0792 | 61.2848 | 103.235 | 101.511 |
| 12 | 34.94 | 59.3136 | 30.8928 | 29.0607 | 35.0787 | 63.084 | 107.463 | 102.968 |
| 13 | 35.0815 | 60.8943 | 119.824 | 110.353 | 35.0808 | 60.0052 | 124.427 | 102.28 |
| 14 | 35.0714 | 63.647 | 115.119 | 101.234 | 34.401 | 60.7519 | 107.747 | 103.273 |
| 15 | 35.0785 | 59.5327 | 117.79 | 106.736 | 34.3965 | 59.6553 | 125.768 | 106.845 |
| 16 | 35.0736 | 59.6032 | 113.938 | 106.101 | 34.0274 | 60.3798 | 119.878 | 106.548 |

This is the table of values I got after running through the project.

The following is the graph made from the data of the table above.

From looking at the graph above, the main thing you can see performance wise is the great shift from NUM 12 to NUM 13 when using 4 and 8 threads. The performance goes up by at least x5 for both thread lines. It seems most likely this is where the false sharing has stopped and the threads start reaching peak performance.

I also see that when using 2 threads the performance starts growing much sooner but never gets to the peak performance as much as 4 and 8 threads. So it must have optimized using all the cache lines it can around NUM 3.

Then with using only 1 thread it’s noticeable that with any number of padding it never truly optimizing. This is because with one thread it can’t take advantage of any parallelizable.

The nice thing I realized it that you can get a nice performance with just 4 threads so you if you really wanted you shouldn’t need to use anything more than 4 threads.